Sensor Planning for Object Pose Estimation and Identification

Jeremy Ma (jerma@caltech.edu), and Joel Burdick (jwb@robotics.caltech.edu)
California Institute of Technology, Pasadena, CA

***Abstract* — This paper proposes a novel approach to sensor planning for simultaneous object identification and 3D pose estimation. We consider the problem of determining the *next-best-view* for a movable sensor (or an autonomous agent) to identify an unknown object from among a database of known object models. We use an information theoretic approach to define a metric (based on the difference between the current and expected model entropy) that guides the selection of the optimal control action. We present a generalized algorithm that can be used in sensor planning for object identification and pose estimation. Experimental results are also presented to validate the proposed algorithm.**

I. INTRODUCTION

This paper considers the combined problem of sensor planning for object identification *with* 3D pose estimation. Our approach selects optimal control actions for sensor placement based on an information gain metric and the currently estimated poses for all possible models. The paper is organized as follows: Section II overviews related work and how our approach differs from previous work. Section III introduces the information gain metric to be optimized and the expressions that form our algorithm. Section IV presents experimental results to verify and illustrate the method.

II. RELATED WORK

The problem of sensor planning for object identification has been extensively investigated. In this "*active recognition*" problem [1][2] the goal is to determine the movement actions needed to gather enough evidence to disambiguate initial object hypotheses. The use of information theoretic concepts has been investigated as a possible solution framework for this problem. In [3], the authors use entropy as their optimization metric. Their *entropy maps*, which are computed on the surface of a discretized sphere, are generated offline from sets of training views captured for all objects in the database. At run time, objects are recognized by selecting the most informative view from the precomputed entropy maps. However, their approach requires a rather tedious and time consuming process to generate the entropy maps, and handles only static objects.

Paletta *et al.* [4] present a Bayesian fusion approach that incorporates the temporal context of observations by integrating multiple recognition results. They develop a view-planning scheme for active recognition that minimizes the expected entropy loss through a Markov Decision Process. Their appearance-based feature vectors lie in eigenspace and are constructed by Karhunen-Loeve expansion [5]. While their real-time algorithm, described in [6], shows interesting results for object classification using Bayesian sequential recognition,

Fig. 1. The setup of our experiment, where the mobile agent is confined to a circular path about the object, which is also mobile.

it cannot track object model pose. Our work differs in that while we use the same cost metric to optimize (entropy loss), our Bayesian analysis integrates object pose estimatation into the optimal control action calculation.

Like [4], Denzler and Brown [7] choose mutual-information as their cost metric and present a similar sequential decision-making process for choosing both optimal gaze-control inputs and viewpoint action selections. Monte Carlo sampling is employed to approximate an analytical solution, which partly adds to the computational complexity of their approach but yields rather accurate results for their chosen features – which are the same as [4]. While we also employ Monte Carlo sampling, computational complexity problems are avoided via a set of approximations. More recently, the authors in [8] present a sequential Bayesian method for active object recognition using an upper bound on the differential entropy as a cost metric. While their use of Gaussian mixture model approximations to prior and posterior distributions of the state variable allows for fast parametric updates, 3D pose estimation and object tracking capabilities are not explored.

III. FORMULATION

A. Problem Statement

Consider an autonomous robot equipped with a stereo camera pair which can access a database of known objects, $\mathscr{M} = \{M_1, M_2, \ldots\}$, that may exist in its environment. Now suppose that one of the objects in the database is presented to the robot, which is tasked with identifying the object and estimating its 3D pose. The object need not be stationary. The question then arises: if the robot is unable to identify

978-1-4244-4778-7/09/$25.00 ©2009 IEEE

the correct model in the initial view of the object, how should it optimally move to identify the object?

B. Approach

For a given model M_i at time t_k, we suppose the object's state, $\mathbf{X}_{k,i}$, and the robot's sparse stereo measurements, $\mathbf{D}_k$ (which includes additional feature descriptors tagged to each stereo point to facilitate in subsequent feature correspondences, *e.g.*, SIFT features, Harris corners, etc.), are governed by the following discrete-time system and measurement models:

$$\mathbf{X}_{k,i} = \mathbf{A}(\mathbf{X}_{k-1,i}) + \mathbf{B}(\mathbf{u}_{k-1}) + \eta \quad (1)$$

$$\mathbf{D}_k = \mathbf{C}(\mathbf{X}_{k,i}) + \xi \quad (2)$$

where $\mathbf{u}_k$ is the known robot control input at time t_k, and η, ξ are white Gaussian noise. $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ may be linear or non-linear functions. Let $\mathbf{D}_{1:k}$ denote the set of data measured from time t_1 to time t_k, and similarly let $\mathbf{u}_{1:k}$ denote the control inputs executed during $[t_1, t_k]$.

In determining the optimal control action to execute for the next-best-view, $\mathbf{u}^*$, we choose an information gain utility function, previously used by [4] and similar to the metric chosen by [7] which has been shown by both authors to yield promising results:

$$\boxed{I_{\mathbf{u}_k} = H(M|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}) - E_{\mathbf{D}_{k+1}}\left[H(M|\mathbf{D}_{1:k+1}, \mathbf{u}_{1:k})\right]}$$

and the optimal control action maximizes the expected information gain:

$$\mathbf{u}^* = \underset{\mathbf{u}_k}{argmax}\, I_{\mathbf{u}_k} \quad (3)$$

The term $H(M|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1})$ is the conditional entropy of model M, conditioned on the data acquired up to t_k and the control actions up to t_{k-1}. $E_{\mathbf{D}_{k+1}}[H(M|\mathbf{D}_{1:k+1}, \mathbf{u}_{1:k})]$ is the expected conditional entropy, with the expectation taken over the future data at the next timestep, i.e. $\mathbf{D}_{k+1}$, after hypothetical control $\mathbf{u}_k$ is applied.

The remainder of this section expands the expression for $I_{\mathbf{u}_k}$ according to the novel conditions of our problem. We begin with the core entropy terms:

$$H(M|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}) = - \sum_{i=0} P(M_i|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}) \log P(M_i|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}) \quad (4)$$

$$H(M|\mathbf{D}_{1:k+1}, \mathbf{u}_{1:k}) \triangleq H^+ = - \sum_{i=0} P(M_i|\mathbf{D}_{1:k+1}, \mathbf{u}_{1:k}) \log P(M_i|\mathbf{D}_{1:k+1}, \mathbf{u}_{1:k}) \quad (5)$$

$$E_{\mathbf{D}_{k+1}}\left[H^+\right] = \int H^+ \cdot p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k}) d\mathbf{D}_{k+1} \quad (6)$$

Calculation of Eq.s (4),(5), and (6) require the evaluation of terms: $P(M_i|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1})$, $P(M_i|\mathbf{D}_{1:k+1}, \mathbf{u}_{1:k})$, and $p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k})$. These terms can be translated into computable expressions with appropriate applications of Bayes Rule:

$$P(M_i|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}) = \frac{\overbrace{p(\mathbf{D}_k|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i)}^{I} \cdot \overbrace{P(M_i|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1})}^{II}}{\sum_{j=0} p(\mathbf{D}_k|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_j) P(M_j|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1})} \quad (7)$$

$$P(M_i|\mathbf{D}_{1:k+1}, \mathbf{u}_{1:k}) = \frac{\overbrace{p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k}, M_i)}^{III} \cdot \overbrace{P(M_i|\mathbf{D}_{1:k}, \mathbf{u}_{1:k})}^{IV}}{\sum_{j=0} p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k}, M_j) P(M_j|\mathbf{D}_{1:k}, \mathbf{u}_{1:k})} \quad (8)$$

$$p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k}) = \sum_{i=0} p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k}, M_i) P(M_i|\mathbf{D}_{1:k}, \mathbf{u}_{1:k}) \quad (9)$$

where I, II, III, and IV are the remaining terms needing further development. Close inspection of these terms shows that III & IV have an identical form to I & II, with a simple index shift; thus expressions for I & II will suffice to define III & IV.

C. Bayes Filter and Model Probabilities

Term I can be rewritten by marginalizing over the state of the i^{th} model, $\mathbf{X}_{k,i}$:

$$p(\mathbf{D}_k|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i) = \int p(\mathbf{D}_k|\mathbf{D}_{1:k-1}, \mathbf{X}_{k,i}, \mathbf{u}_{1:k-1}, M_i) \cdot p(\mathbf{X}_{k,i}|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i)\, d\mathbf{X}_{k,i} \quad (10)$$

where $p(\mathbf{D}_k|\mathbf{D}_{1:k-1}, \mathbf{X}_{k,i}, \mathbf{u}_{1:k-1}, M_i)$ will come from the measurement model (Eq.2). The second term of the integral will come from the prediction step of a Bayes' Filter, applied to object pose estimation and tracking. Recall that a Bayes' Filter recursively cycles between two steps:

DYNAMIC PREDICTION:

$$p(\mathbf{X}_{k,i}|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i) = \int p(\mathbf{X}_{k,i}|\mathbf{X}_{k-1,i}, \mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i) \cdot p(\mathbf{X}_{k-1,i}|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i) d\mathbf{X}_{k-1,i} \quad (11)$$

MEASUREMENT UPDATE:

$$p(\mathbf{X}_{k,i}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}, M_i) = \eta_0 p(\mathbf{D}_k|\mathbf{X}_{k,i}, \mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i)\, p(\mathbf{X}_k|\mathbf{D}_{1:k-1}, \mathbf{u}_{1:k-1}, M_i) \quad (12)$$

where η_0 is a normalizing constant. The Bayes Filter terms can be developed from Eq.s (1) and (2) if the following assumption is made:

$$p(\mathbf{X}_{k,i}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k}, M_i) = p(\mathbf{X}_k|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}, M_i)\ . \quad (13)$$

Eq. (13) presumes that the current state, $\mathbf{X}_{k,i}$, is only dependent on data up until the current timestep and the control actions up until the *previous* timestep. This assumption is valid since

the current control action, $\mathbf{u}_k$, does no effect the *current* state, and can thus be omitted.

An argument similar to that of Eq.13 holds for term II:

$$P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k}) = P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k-1}) \tag{14}$$

Thus, inspection of Eq. (7) shows that $P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k-1})$ can be recursively calculated provided that the model probabilities are set to a priori values at t_0, *i.e.*:

$$P(M_i|\mathbf{D}_0,\mathbf{u}_0) = P_{o,i} \tag{15}$$

D. Monte Carlo Sampling

Though terms III and IV can be solved for in a like manner as I and II, they require further analysis. Note that the expectation in Eq. 6 is taken with respect to future data, $\mathbf{D}_{k+1}$, conditioned only on past data measurements and control inputs. Marginalizing over the models and simplifying the results leads to:

$$\begin{aligned} &E_{\mathbf{D}_{k+1}}[H^+] \qquad (16)\\ &= \int H^+ p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k},\mathbf{u}_{1:k})\, d\mathbf{D}_{k+1} \\ &= \int H^+ \sum_i p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k},\mathbf{u}_{1:k},M_i) P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k})\, d\mathbf{D}_{k+1} \\ &= \sum_i P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k}) \int H \cdot p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k},\mathbf{u}_{1:k},M_i) d\mathbf{D}_{k+1} \end{aligned}$$

The remaining integral is analytically intractable due to the high dimension of the space of future measurements. It can be approximated, however, using Monte Carlo sampling. With the definitions:

$$\begin{aligned} g(\mathbf{D}_{k+1}) &= H^+ \qquad (17)\\ f(\mathbf{D}_{k+1}) &= p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k},\mathbf{u}_{1:k},M_i) \qquad (18) \end{aligned}$$

Eq. (16) reduces to:

$$\begin{aligned} &E_{\mathbf{D}_{k+1}}[H^+] \qquad (19)\\ &= \sum_i P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k}) \int g(\mathbf{D}_{k+1}) \cdot f(\mathbf{D}_{k+1}) d\mathbf{D}_{k+1} \\ &\approx \sum_i P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k}) \frac{1}{N} \sum_n^N g(\tilde{\mathbf{D}}) \end{aligned}$$

where $\tilde{\mathbf{D}} \sim f(\mathbf{D}_{k+1})$. Thus, $E_{\mathbf{D}_{k+1}}[H^+]$ is calculated as a weighted sum of all model entropies with the weight chosen as the model probability. The model entropies are found by simulating N future measurements from model M_i for which N model entropies are then calculated and averaged to yield a measure of the expected information associated with the control action, $\mathbf{u}_k$. Details on how future measurements are simulated are discussed in Section IV.

E. Algorithm

Algorithm 1 details how to implement the above analysis in a recursive form. The following additional definitions are used in Algorithm 1 to simplify the expressions:

$$\begin{aligned} \underline{bel}_{k,i} &= p(\mathbf{X}_{k,i}|\mathbf{D}_{1:k},\mathbf{u}_{1:k-1},M_i) \\ \overline{bel}_{k,i} &= p(\mathbf{X}_{k,i}|\mathbf{D}_{1:k-1},\mathbf{u}_{1:k-1},M_i) \\ \overline{p}_{k,i} &= P(M_i|\mathbf{D}_{1:k},\mathbf{u}_{1:k-1}) \\ \overline{H}_k &= H(M|\mathbf{D}_{1:k},\mathbf{u}_{1:k-1}) \end{aligned}$$

Algorithm 1 Next-Best View

1: acquire data D_k from sensor
2: $I_{max} = 0$
3: $\mathbf{u}^* = NULL$
4: $\overline{H}_k = calcCurrentEntropy(\mathbf{X}_{k-1,i},\mathbf{D}_{1:k},\mathbf{u}_{1:k-1},M)$
5: $\mathbf{U}^+ = calcPossibleControlActions(\mathbf{X}_{k-1,i},\mathbf{u}_{1:k-1})$
6: **for** all $\mathbf{u}_m^+ \in \mathbf{U}^+$ **do**
7: $H_m^+ =$ $calcExpectedEntropy(\mathbf{X}_{k-1,i},\mathbf{D}_{1:k},\mathbf{u}_{1:k-1},\mathbf{u}_m^+,M)$
8: $I_{gain} = \overline{H}_k - H_m^+$
9: **if** $I_{gain} > I_{max}$ **then**
10: $I_{max} = I_{gain}$
11: $\mathbf{u}^* = \mathbf{u}_m^+$
12: **end if**
13: **end for**
14: execute $\mathbf{u}^*$ and repeat

During each algorithm cycle, the current model entropy is calculated and the expected model entropy is estimated for all possible future actions, $\mathbf{u}_m^+$. The future action which yields the greatest information gain is selected and executed.

The function *calcPossibleControlActions* depends on the nature of the actuating hardware. Additional criteria, such as the cost of control actions (e.g. power consumption), can be added, but we omit those details.

F 1 $calcCurrentEntropy(\mathbf{X}_{k-1,i},\mathbf{D}_{1:k},\mathbf{u}_{1:k-1},M)$

1: $\overline{H}_k = 0$
2: **for** all M_i **do**
3: $\overline{bel}_{k,i} =$ $\int p(\mathbf{X}_{k,i}|\mathbf{X}_{k-1,i},\mathbf{D}_{1:k-1},\mathbf{u}_{1:k-1},M_i)\, bel_{k-1,i}\, d\mathbf{X}_{k-1,i}$
4: $bel_{k,i} = p(\mathbf{D}_k|\mathbf{D}_{1:k-1},\mathbf{X}_{k,i},\mathbf{u}_{1:k-1},M_i)\, \overline{bel}_{k,i} \cdot \eta_0$
5: $p(\mathbf{D}_k|\mathbf{D}_{1:k-1},\mathbf{u}_{1:k-1},M_i) =$ $\int p(\mathbf{D}_k|\mathbf{D}_{1:k-1},\mathbf{X}_{k,i},u_{1:k-1},M_i)\, \overline{bel}_{k,i} d\mathbf{X}_{k,i}$
6: $\overline{p}_{k,i} = p(\mathbf{D}_k|\mathbf{D}_{1:k-1},\mathbf{u}_{1:k-1},M_i) \cdot \overline{p}_{k-1,i} \cdot \eta_1$
7: $\overline{H}_k = -\overline{p}_{k,i} \log \overline{p}_{k,i} + \overline{H}_k$
8: **end for**
9: return $\overline{H}_k$

The function *calcCurrentEntropy* (see Function **F1**) computes the current model entropy based on the available data and robot state. This function also updates the following (internal) terms: $\overline{bel}_{k,i}$, $bel_{k,i} \rightarrow \mathbf{X}_{k,i}$, and $\overline{p}_{k,i}$. Lines 3-4 of **F1** implement the Bayes Filter on the robot state, as needed to calculate the model probabilities. Hence, an updated object state estimate for each potential model is calculated as a byproduct.

The function *calcExpectedEntropy* (see Function **F2**) computes the expected model entropy for a proposed control input, $\mathbf{u}_m^+$. It makes use of the $bel_{k,i}$, $\overline{p}_{k,i}$, and $\mathbf{X}_{k,i}$ terms computed

in *calcCurrentEntropy* to predict the future object pose for all possible models (Lines 4-5). Once calculated, Monte Carlo sampling is then performed for each model to extract expected data measurements and to determine model entropies (Lines 6-16). The resultant entropies are averaged over all models to yield an expected model entropy. The normalizing terms η_0, η_1, and η_2 in both Function **F1** and Function **F2** ensure the associated probability values sum/integrate to 1.

F 2 *calcExpectedEntropy*$(\mathbf{X}_{k-1,i}, \mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+, M)$

1: $H_m^+ = 0$
2: **for** all M_i **do**
3: $\quad G = 0$
4: $\quad \overline{bel}_{k+1,i} = \int p(\mathbf{X}_{k+1,i}|\mathbf{X}_{k,i}, \mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+, M_i)\, bel_{k,i} d\mathbf{X}_{k,i}$
5: $\quad p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+, M_i) = \int p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{X}_{k+1,i}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+, M_i)\, \overline{bel}_{k+1,i} d\mathbf{X}_{k+1,i}$
6: $\quad$ **for** $n = 1$ to N **do**
7: $\quad\quad h = 0$
8: $\quad\quad$ sample $\tilde{\mathbf{D}} \sim p(\mathbf{D}_{k+1}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+, M_i)$
9: $\quad\quad$ **for** all M_j **do**
10: $\quad\quad\quad P(M_j|\mathbf{D}_{1:k}, \tilde{\mathbf{D}}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+) = p(\tilde{\mathbf{D}}|\mathbf{D}_{1:k}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+, M_j) \cdot \overline{p}_{k,j} \cdot \eta_2$
11: $\quad\quad\quad h = -P(M_j|\mathbf{D}_{1:k}, \tilde{\mathbf{D}}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+) \cdot \log P(M_j|\mathbf{D}_{1:k}, \tilde{\mathbf{D}}, \mathbf{u}_{1:k-1}, \mathbf{u}_m^+) + h$
12: $\quad\quad$ **end for**
13: $\quad\quad G = h + G$
14: $\quad$ **end for**
15: $\quad G = \frac{1}{N} \cdot G$
16: $\quad H_m^+ = G \cdot \overline{p}_{k,i} + H_m^+$
17: **end for**
18: return H_m^+

IV. EXPERIMENTAL RESULTS

A. Experiment Formulation and Setup

We now apply the algorithm described above to an experiment involving a two-wheeled, non-holonomic robot tasked with visually identifying an unknown mobile object (whose movements are operator controlled) from a finite database of known models. For simplicity, we confine the mobile agent to move on a ring initially centered about the unknown moving object. Control actions of the agent are restricted to a discrete set of wheel speeds and turnrates corresponding to locations along the circumference of the ring. This reduces the agent's pose to a single degree-of-freedom, which is parametrized by the angle of its position on the ring circumference (where the angle is measured with respect to a global coordinate frame placed at the center of the circle). The agent is equipped with onboard wheel odometry for localization and a fixed stereo camera head for sensing. SIFT features [9] are used augmented with 3D sparse stereo data, and object identification is done using the Best-Bin-First matching schema of [9].

The experimental system (see Figure 1) used a PointGrey^{TM} BumbleBee2 stereo color camera (downsampled to 320x240 resolution) mounted on an Evolution Robotics, Inc.^{TM}, ER-1 mobile robot. Wheel odometry was used as a crude measure of robot pose estimation. Computations were performed on a laptop (Intel Pentium(R) M 1.86GHz processor) running Linux. The algorithm was written in C/C++.

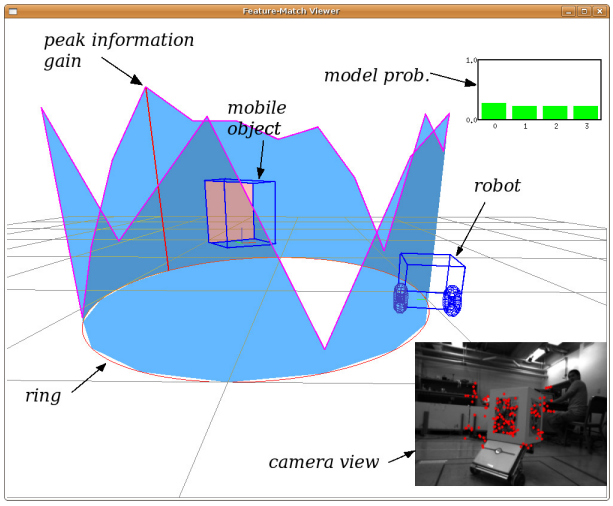

Fig. 3. A screenshot of the algorithm's visual interface. Shown is the information gain for various future robot locations (limited to a circle passing through robot's current pose). The optimal action moves to the peak of the information gain metric, which brings the discriminating face (shown in pink) into view.

For the mobile object, the database consisted of 4 models of a box attached to a moveable base. Each lateral side of the box was marked with a distinct pattern (a magazine cutout), and it was decided that all 4 models would share 3 identical faces, with only the last face being different for each model. The database of features for each model was generated during an off-line "training phase" in which the robot was allowed to learn each object from varying "viewpoints".

In implementing the proposed algorithm, we assumed Gaussian white noise in both the motion and measurement model. Sampling of future data was done through a series of look up tables for each model, catalogued according to object pose $(x, y, z, \alpha, \beta, \gamma)$ relative to the camera reference frame.

B. Results

In the first trial, Model 2 was placed in the center of a ring whose size corresponded to the training ring. The object's movements were controlled by a remote operator who strove to move the target so as to prevent a direct observation of the discriminating face by the agent. The purpose of this strategy was merely to test the limits of our proposed algorithm and the ability of the agent to track, estimate, and plan a sensing action to identify the object.

The top two rows of Figure 2 show the position parameters $(x, y, z, \alpha, \beta, \gamma)$ estimated by our algorithm plotted as a function of time. Though poses are estimated for all database models, only the parameters for the true model are shown. Plotted against the estimated parameters is the model object reference pose as determined by wheel odometery. Considering that the

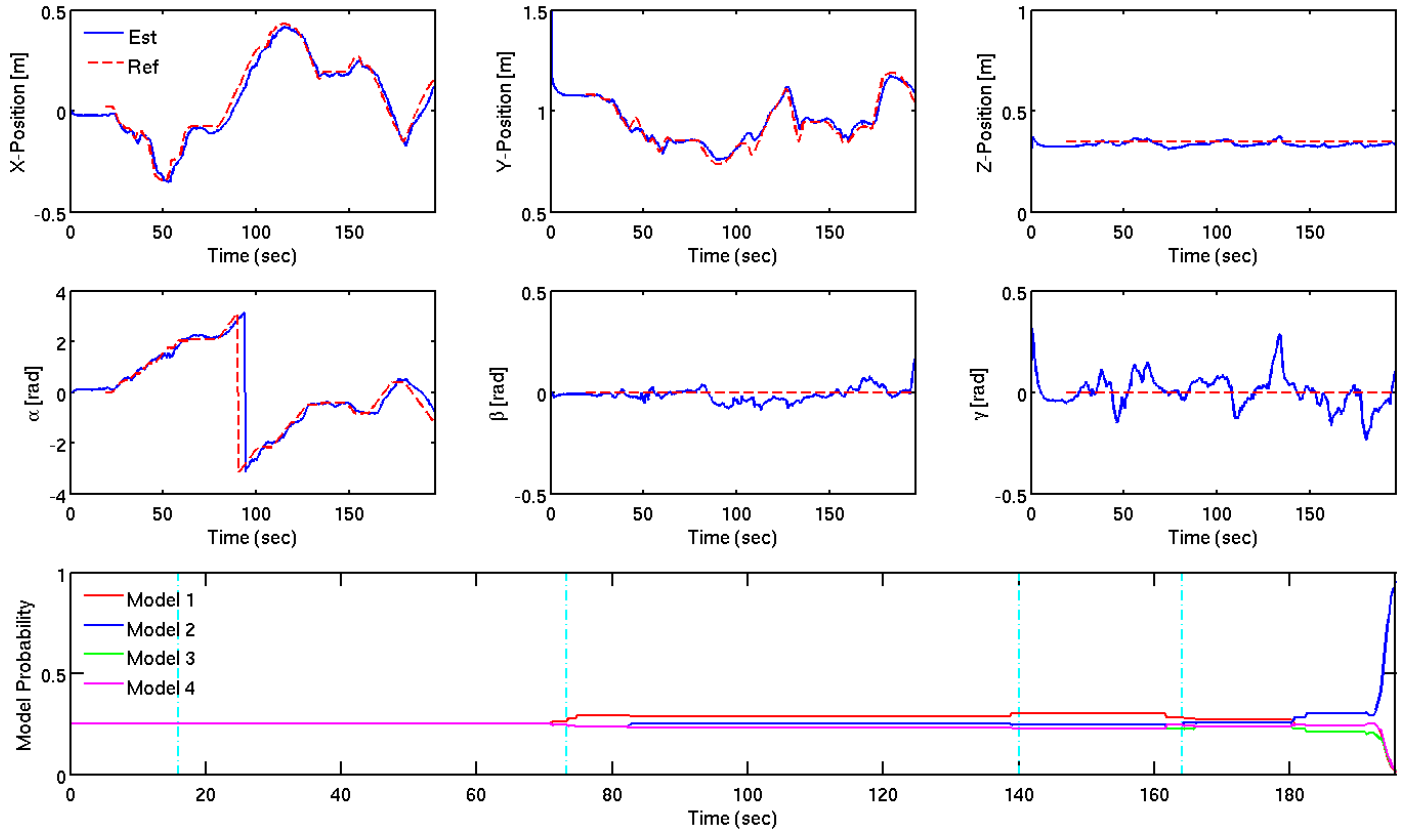

Fig. 2. The position estimates of the object (as defined in a global reference frame) are shown in the top row and the orientation estimates are shown in the middle row. The robot estimated pose using our proposed algorithm is shown in the solid-blue line and the reference pose is shown in the dotted-red line. The last row shows the model probability estimates for all models, plotted against time. The vertical dotted lines indicate instances of when the information gain calculations were executed for sensor planning.

algorithm assumes a simple random walk model for the object, the results show that it does a fairly good job of tracking. Note however the noise associated with the roll angle estimate in the far right plot of the second row – a variance of roughly $0.3rad$. This is largely attributed to poor stereo projection of certain features and also a small set of feature mismatches.

The bottom row of Figure 2 shows the calculated model probabilities for each model during the same trial. Note that up until $t = 180s$, the model probabilities are roughly equal at 0.25. This is expected since all models share the same three faces, and until the discriminating model face is observed (which happens at $t = 180s$), each model is equally likely of being the unknown object. Note that once that face is observed, the model probability of the correct model increases rapidly to 1.0 while the other probabilities decrease to 0.0. The dotted vertical lines correspond to the time instances when the information gain is calculated and the optimal action chosen. Figure 3 illustrates a screenshot of the system's visual interface taken at such an updating event. It is important to note that the information gain calculations are not performed during each timestep, but are executed once the robot has completed the prior sensing action.

Subsequent experiment trials were carried out for the other objects in the database. Figure 4 shows the results of those trials (with the pose estimate plot comparisons omitted for brevity), with the true models indicated by the asterisk. As can be seen in the figure, the algorithm accurately identifies the true models in each trial run and shows consistent behavior across all trials.

V. CONCLUSIONS AND FUTURE WORK

We proposed a novel approach to sensor planning for simultaneous object identification and 3D pose estimation. Our results show that the algorithm can be applied to a mobile robot tasked with identifying and tracking an unknown mobile object. In future work, we hope to extend the algorithm to many more objects and incorporate unconstrainted motion of the agent (*i.e.* no longer confined to motion on a ring).

REFERENCES

[1] F. G. Callari and F. P. Ferrie, "Autonomous recognition: Driven by ambiguity," in *Proc. Conf. on Computer Vision and Pattern Recognition*, 1996, pp. 701–707.

[2] E. Rivlin, Y. Aloimonos, and A. Rosenfeld, "Purposive recognition: An active and qualitative approach," in *Proc. SPIE (Int. Society for Optical Engineering)*, 1991, pp. 225–240.

[3] T. Arbel and F. P. Ferrie, "Entropy-based gaze planning," in *Proc. 2nd IEEE Workshop on Perception for Mobile Agents*, 1999.

[4] L. Paletta, M. Prantl, and A. Pinz, "Learning temporal context in active object recognition using bayesian analysis," in *Proc. 15th Int. Conf. Pattern Recognition*, vol. 1, 2000, pp. 695–699.

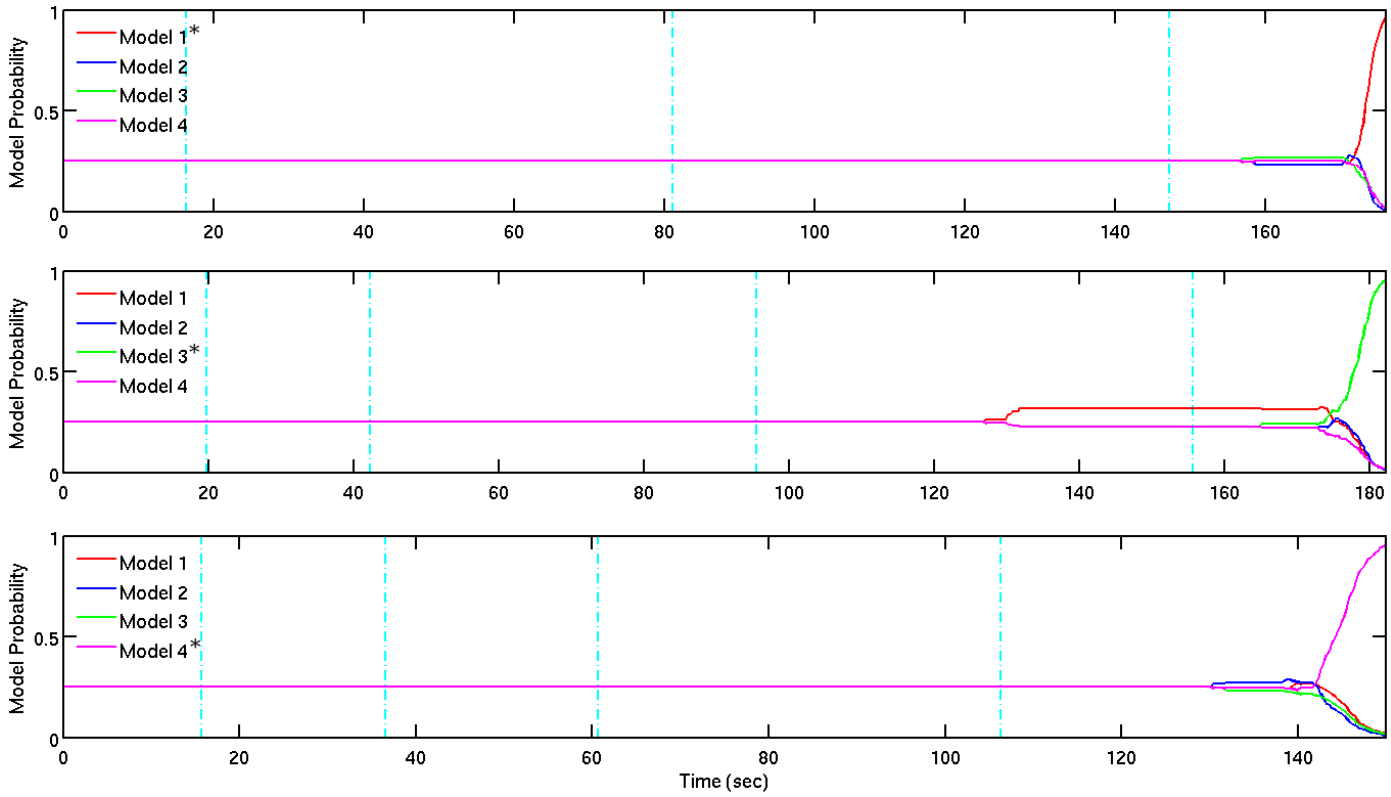

Fig. 4. Other trials were also considered for different models. Shown are those model probability estimates for each trial plotted against time. The true models are indicated with the asterisk. As can be seen, the algorithm accurately identifies the true model in each case. The vertical dotted lines indicate instances of when information gain calculations were executed for sensor planning.

[5] H. Murase and S. Nayar, "Visual learning and recognition of 3-d objects from appearance," vol. 14, pp. 5–24, 1995.

[6] L. Paletta and A. Pinz, "Active object recognition by view integration and reinforcement learning," *Robotics and Autonomous Systems*, vol. 31, pp. 71–86, 2000.

[7] J. Denzler and C. Brown, "Information theoretic sensor data selection for active object recognition and state estimation," *IEEE Trans. Pattern Anal. Machine Intel.*, vol. 24, pp. 145–157, 2002.

[8] R. Eidenberger, T. Grundmann, W. Feiten, and R. Zoellner, "Fast parametric viewpoint estimation for active object detection," in *Proc. Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, 2008, pp. 309–314.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Computer Vision*, vol. 60, pp. 91–110, 2004.